

Transforming Hidden Vector Encryption Schemes from Composite to Prime Order Groups

Kwangsu Lee*

Abstract

Predicate encryption is a new type of public key encryption that enables searches on encrypted data. By using predicate encryption, we can search keywords or attributes on encrypted data without decrypting ciphertexts. Hidden vector encryption (HVE) is a special kind of predicate encryption. HVE supports the evaluation of conjunctive equality, comparison, and subset operations between attributes in ciphertexts and attributes in tokens. In this paper, we construct efficient HVE schemes in prime order bilinear groups derived from previous HVE schemes in composite order bilinear groups, and prove their selective security under simple assumptions. To achieve this result, we present a conversion method that transforms HVE schemes from composite order bilinear groups into prime order bilinear groups. Our method supports any types of prime order bilinear groups and uses simple assumptions.

Keywords: Searchable encryption, Predicate encryption, Hidden vector encryption, Conversion method, Bilinear maps.

*Sejong University, Seoul, Korea. Email: kwangsu@sejong.ac.kr.

1 Introduction

Searchable public key encryption is a new type of public key encryption (PKE) that enables efficient searching on encrypted data [3]. In PKE, if an agent A wants to search on encrypted data for a user B , he should first decrypt ciphertexts using the private key SK of the user B . This simple method has a problem that the agent requires the user's private key. In searchable public key encryption, a ciphertext is associated with keywords or attributes, and a user can generate a token for searching from the user's private key. That is, an agent A performs searches on encrypted data using the token TK that is related with keywords or attributes instead of using the private key SK . By using searchable public key encryption, it is possible to build interesting systems like privacy preserving mail gateway systems [3], secure audit log systems [25], network audit log systems [21], and credit card payment gateway systems [7].

Predicate encryption (PE) is a generalization of searchable public key encryption [7, 14]. In PE, a ciphertext is associated with an attribute x , and a token is associated with a predicate f . At first, a sender creates a ciphertext that is associated with an attribute x , and an agent receives a token that corresponds to a predicate f from a receiver. If $f(x) = 1$, then the agent can decrypt ciphertexts that are related with x . Otherwise, that is $f(x) = 0$, then the agent cannot get any information except that $f(x) = 0$. That is, PE provides both *message hiding* and *attribute hiding* properties. Hidden vector encryption (HVE) is a special kind of PE [7]. In HVE, a ciphertext and a token are associated with attribute vectors \mathbf{x}, \mathbf{y} respectively, and the attribute vector for the token contains a special wild card attribute. If each attribute of a ciphertext is equal with the attribute of a token except the wild card attribute, then the predicate $f_{\mathbf{y}}(\mathbf{x})$ is satisfied. HVE supports the evaluation of predicates such that conjunctive equality, conjunctive subset, and conjunctive comparison.

Many HVE schemes were originally proposed in composite order bilinear groups [7, 16, 22]. To improve the efficiency of HVE schemes, HVE schemes in prime order bilinear groups are required. Although many HVE schemes in prime order groups were constructed from scratch [13, 18, 19], we would like to easily obtain HVE schemes in prime order groups from previous schemes in composite order groups. The previous conversion methods that convert cryptographic schemes from composite order to prime order bilinear groups are Freeman's method [9] and Ducas' method [8]. The method of Ducas is that random blinding elements in ciphertexts can be eliminated in asymmetric bilinear groups of prime order since the decisional Diffie-Hellman (DDH) assumption holds in asymmetric bilinear groups. The method of Freeman is that product groups and vector orthogonality provide the subgroup decision assumption and the subgroup orthogonality property in prime order bilinear groups, respectively. The merit of this method is that it can convert many cryptographic schemes from bilinear groups of composite order to asymmetric bilinear groups of prime order. The demerits of this method are that the converted scheme only works in asymmetric bilinear groups and the security of the scheme is proven under complex assumptions.

1.1 Our Results

In this paper, we present a new conversion method that transforms HVE schemes from composite order bilinear groups into prime order bilinear groups.

Our conversion method is similar to the conversion method of Freeman [9] since it uses product groups and vector orthogonality, but ours has the following three differences. The first difference is that Freeman's method is related to the subgroup decision (SD) assumption in prime order bilinear groups, whereas our method is not related to the SD assumption. The second difference is that Freeman's method only works in asymmetric bilinear groups of prime order, whereas our method works in any bilinear groups of prime order. The third difference is that cryptographic schemes that are converted from Freeman's method use

complex assumptions that depend on complex basis vectors, whereas HVE schemes that are converted from our method use simple assumptions that are independent of basis vectors.

By using our conversion method, we first convert the HVE scheme of Boneh and Waters [7] in composite order bilinear groups into an HVE scheme in symmetric bilinear groups of prime order. We then prove the converted HVE scheme is selectively secure under the decisional bilinear Diffie-Hellman (DBDH) and the parallel 3-party Diffie-Hellman (P3DH) assumptions. Next, we also convert the delegatable HVE scheme of Shi and Waters [22] and the efficient HVE scheme of Lee and Lee [16] from composite order bilinear groups to HVE schemes in symmetric bilinear groups of prime order. Finally, we show that the new P3DH assumption holds in generic group model introduced by Shoup.

1.2 Related Work

PE is closely related to functional encryption [6]. In functional encryption, a ciphertext is associated with attributes \mathbf{x} , and a private key is associated with a function f . The main difference between PE and functional encryption is that the computation of a predicate $f(x) \in \{0, 1\}$ is only allowed in PE whereas the computation of any function $f(x)$ is allowed in functional encryption. Identity-based encryption (IBE) is the most simple type of functional encryption, and it provides an equality function for an identity in ciphertexts [4]. Hierarchical IBE (HIBE) is an extension of IBE, and it provides a conjunctive equality function for a hierarchical identity in ciphertexts [11]. Attribute-based encryption (ABE) is also an extension of IBE, and it provides the most general function that consists of AND, OR, NOT, and threshold gates [12].

The first HVE scheme was proposed by Boneh and Waters [7]. After their construction, various HVE schemes were proposed in [8, 16, 22]. A simple HVE scheme can be constructed from a PKE scheme [3, 7, 15]. This method was introduced by Boneh et al. [3] to construct a PKE scheme with keyword search (PEKS) using trapdoor permutations. After that, Boneh and Waters showed that a searchable public key encryption for general predicates also can be constructed from this method [7]. Katz and Yerukhimovich [15] showed that it is possible to construct a PE scheme from a PKE scheme if the number of predicate is less than a polynomial number of a security parameter. The main idea of this method is to use a multiple instances of key-private PKE introduced by Bellare et al. [1]. That is, the public key of searchable public key encryption consists of the public keys of key-private PKE and each instance of public keys is mapped to each predicate. However, this method has a serious problem that the total number of predicates is limited to the polynomial value of a security parameter.

Another HVE scheme can be constructed by extremely generalizing anonymous IBE (AIBE) [7, 8, 13, 16, 19, 22]. This method was introduced by Boneh and Waters [7]. They used the IBE scheme of Boneh and Boyen [2] and composite order bilinear groups to provide the anonymity of ciphertexts. Shi and Waters constructed a delegatable HVE scheme [22]. Lee and Lee constructed an efficient HVE scheme with a constant number of pairing operations [16]. In composite order bilinear groups, the random blinding property using subgroups provides the anonymity of ciphertexts and the orthogonal property among subgroups provides the successful decryption. However, it is inefficient to use composite order bilinear groups since the group order of composite order bilinear groups should be large. To overcome this problem of inefficiency, Freeman presented a general framework that converts cryptographic schemes from composite order bilinear groups to prime order bilinear groups [9]. Ducas also showed that HVE schemes in composite order bilinear groups are easily converted to schemes in prime order bilinear groups [8]. However, these conversion methods result in asymmetric bilinear groups.

Finally, an HVE scheme can be derived from inner-product encryption (IPE) [14, 18, 20]. IPE is a kind of PE and it enable the evaluation of inner-product predicates between the vector of ciphertexts and the vector of tokens. Katz et al. [14] constructed the first IPE scheme under composite order bilinear groups.

Okamoto and Takashima constructed an hierarchical IPE scheme using dual pairing vector spaces [18]. Park proposed an IPE scheme under prime order bilinear groups and proved its security under the well-known assumptions [20]. The main idea of converting an IPE scheme to an HVE scheme is to construct a predicate of conjunctive equality using a predicate of inner product [14].

2 Preliminaries

In this section, we define hidden vector encryption, and introduce bilinear groups of prime order and two complexity assumptions.

2.1 Hidden Vector Encryption

Let Σ be a finite set of attributes and let $*$ be a special symbol not in Σ . Define $\Sigma_* = \Sigma \cup \{*\}$. The star $*$ plays the role of a wild-card or “don’t care” value. For a vector $\vec{\sigma} = (\sigma_1, \dots, \sigma_\ell) \in \Sigma_*^\ell$, we define a predicate $f_{\vec{\sigma}}$ over Σ^ℓ as follows: For $\vec{x} = (x_1, \dots, x_\ell) \in \Sigma^\ell$, it set $f_{\vec{\sigma}}(\vec{x}) = 1$ if $\forall i : (\sigma_i = x_i \text{ or } \sigma_i = *)$, it set $f_{\vec{\sigma}}(\vec{x}) = 0$ otherwise.

Definition 2.1 (Hidden Vector Encryption). An HVE scheme consists of four algorithms **Setup**, **GenToken**, **Encrypt**, and **Query** which are defined as follows:

Setup($1^\lambda, \ell$): The setup algorithm takes as input a security parameter 1^λ and the length parameter ℓ . It outputs a public key PK and a secret key SK .

GenToken($\vec{\sigma}, SK, PK$): The token generation algorithm takes as input a vector $\vec{\sigma} = (\sigma_1, \dots, \sigma_\ell) \in \Sigma_*^\ell$ that corresponds to a predicate $f_{\vec{\sigma}}$, the secret key SK and the public key PK . It outputs a token $TK_{\vec{\sigma}}$ for the vector $\vec{\sigma}$.

Encrypt(\vec{x}, M, PK): The encrypt algorithm takes as input a vector $\vec{x} = (x_1, \dots, x_\ell) \in \Sigma^\ell$, a message $M \in \mathcal{M}$, and the public key PK . It outputs a ciphertext CT for \vec{x} and M .

Query($CT, TK_{\vec{\sigma}}, PK$): The query algorithm takes as input a ciphertext CT , a token $TK_{\vec{\sigma}}$ for a vector $\vec{\sigma}$ that corresponds to a predicate $f_{\vec{\sigma}}$, and the public key PK . It outputs M if $f_{\vec{\sigma}}(\vec{x}) = 1$ or outputs \perp otherwise.

The scheme should satisfy the following correctness property: For all $\vec{x} \in \Sigma^\ell$, $M \in \mathcal{M}$, $\vec{\sigma} \in \Sigma_*^\ell$, let $(PK, SK) \leftarrow \mathbf{Setup}(1^\lambda, \ell)$, $CT \leftarrow \mathbf{Encrypt}(\vec{x}, M, PK)$, and $TK_{\vec{\sigma}} \leftarrow \mathbf{GenToken}(\vec{\sigma}, SK, PK)$.

- If $f_{\vec{\sigma}}(\vec{x}) = 1$, then $\mathbf{Query}(CT, TK_{\vec{\sigma}}, PK) = M$.
- If $f_{\vec{\sigma}}(\vec{x}) = 0$, then $\mathbf{Query}(CT, TK_{\vec{\sigma}}, PK) = \perp$ with all but negligible probability.

Definition 2.2 (Selective Security). The selective security of HVE is defined as the following game between a challenger \mathcal{C} and an adversary \mathcal{A} :

1. **Init**: \mathcal{A} submits two vectors $\vec{x}_0, \vec{x}_1 \in \Sigma^\ell$.
2. **Setup**: \mathcal{C} runs the setup algorithm and keeps the secret key SK to itself, then it gives the public key PK to \mathcal{A} .

3. **Query 1:** \mathcal{A} adaptively requests a polynomial number of tokens for vectors $\vec{\sigma}_1, \dots, \vec{\sigma}_{q_1}$ that correspond to predicates $f_{\vec{\sigma}_1}, \dots, f_{\vec{\sigma}_{q_1}}$ subject to the restriction that $f_{\vec{\sigma}_i}(\vec{x}_0) = f_{\vec{\sigma}_i}(\vec{x}_1)$ for all i . In responses, \mathcal{C} gives the corresponding tokens $TK_{\vec{\sigma}_i}$ to \mathcal{A} .
4. **Challenge:** \mathcal{A} submits two messages M_0, M_1 subject to the restriction that if there is an index i such that $f_{\vec{\sigma}_i}(\vec{x}_0) = f_{\vec{\sigma}_i}(\vec{x}_1) = 1$ then $M_0 = M_1$. \mathcal{C} chooses a random coin γ and gives a ciphertext CT of $(\vec{x}_\gamma, M_\gamma)$ to \mathcal{A} .
5. **Query 2:** \mathcal{A} continues to request tokens for vectors $\vec{\sigma}_{q_1+1}, \dots, \vec{\sigma}_q$ that correspond to predicates $f_{\vec{\sigma}_{q_1+1}}, \dots, f_{\vec{\sigma}_q}$ subject to the two restrictions as before.
6. **Guess:** \mathcal{A} outputs a guess γ' . If $\gamma = \gamma'$, it outputs 0. Otherwise, it outputs 1.

The advantage of \mathcal{A} is defined as $\mathbf{Adv}_{\mathcal{A}}^{HVE}(1^\lambda) = |\Pr[\gamma = \gamma'] - 1/2|$ where the probability is taken over the coin tosses made by \mathcal{A} and \mathcal{C} . We say that an HVE scheme is selectively secure if all probabilistic polynomial-time (PPT) adversaries have at most a negligible advantage in the above game.

2.2 Bilinear Groups of Prime Order

Let \mathbb{G} and \mathbb{G}_T be multiplicative cyclic groups of prime p order. Let g be a generator of \mathbb{G} . The bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ has the following properties:

1. Bilinearity: $\forall u, v \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degeneracy: $\exists g$ such that $e(g, g)$ has order p , that is, $e(g, g)$ is a generator of \mathbb{G}_T .

We say that $(p, \mathbb{G}, \mathbb{G}_T, e)$ are bilinear groups if the group operations in \mathbb{G} and \mathbb{G}_T as well as the bilinear map e are all efficiently computable.

2.3 Complexity Assumptions

We introduce two simple assumptions under prime order bilinear groups. The decisional bilinear Diffie-Hellman assumption was introduced in [4]. The parallel 3-party Diffie-Hellman (P3DH) assumption is newly introduced in this paper.

Assumption 1 (Decisional Bilinear Diffie-Hellman, DBDH). Let $(p, \mathbb{G}, \mathbb{G}_T, e)$ be a description of the bilinear group of prime order p . The DBDH problem is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g, g^a, g^b, g^c) \text{ and } T,$$

decides whether $T = T_0 = e(g, g)^{abc}$ or $T = T_1 = e(g, g)^d$ with random choices of $a, b, c, d \in \mathbb{Z}_p$. The advantage of \mathcal{A} is defined as $\mathbf{Adv}_{\mathcal{A}}^{DBDH}(1^\lambda) = |\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]|$ where the probability is taken over the random choices of $a, b, c, d \in \mathbb{Z}_p$ and the random bits used by \mathcal{A} . We say that the DBDH assumption holds if no PPT algorithm has a non-negligible advantage in solving the above problem.

Assumption 2 (Parallel 3-party Diffie-Hellman, P3DH). Let $(p, \mathbb{G}, \mathbb{G}_T, e)$ be a description of the bilinear group of prime order p . The P3DH problem is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), (g, f), (g^a, f^a), (g^b, f^b), (g^{ab} f^{z_1}, g^{z_1}), (g^{abc} f^{z_2}, g^{z_2})) \text{ and } T,$$

decides whether $T = T_0 = (g^c f^{z_3}, g^{z_3})$ or $T = T_1 = (g^d f^{z_3}, g^{z_3})$ with random choices of $a, b, c, d \in \mathbb{Z}_p$ and $z_1, z_2, z_3 \in \mathbb{Z}_p$. The advantage of \mathcal{A} is defined as $\text{Adv}_{\mathcal{A}}^{\text{P3DH}}(1^\lambda) = |\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]|$ where the probability is taken over the random choices of $a, b, c, d, z_1, z_2, z_3$ and the random bits used by \mathcal{A} . We say that the P3DH assumption holds if no PPT algorithm has a non-negligible advantage in solving the above problem.

Remark 1. The P3DH problem can be modified as follows: given a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), (g, f), (g^a, f^a), (g^b, f^b), (g^{ab} f^{z_1}, g^{z_1}), (g^c f^{z_2}, g^{z_2}))$ and T , decides whether $T = T_0 = (g^{abc} f^{z_3}, g^{z_3})$ or $T = T_1 = (g^d f^{z_3}, g^{z_3})$. However, this modified one is the same as the original one by changing the position of the challenge tuple as $D = ((p, \mathbb{G}, \mathbb{G}_T, e), (g, f), (g^a, f^a), (g^b, f^b), (g^{ab} f^{z_1}, g^{z_1}), T)$ and $T' = (g^c f^{z_2}, g^{z_2})$. Thus, we will use any one of challenge tuple forms for the P3DH assumption.

3 Our Techniques

The basic idea to convert HVE schemes from composite order bilinear groups to prime order bilinear groups is to use bilinear product groups that are extended from bilinear groups using the direct product operation. Bilinear product groups were widely used in dual system encryption of Waters [17, 24], private linear broadcast encryption of Garg et al. [10], and the conversion method of Freeman [9]. The product groups extended from multiplicative cyclic groups represent an exponent as a vector. Thus vector operations in product groups and bilinear product groups should be defined. Definition 3.1 and Definition 3.2 define the vector operations in product groups and bilinear product groups, respectively.

Definition 3.1 (Vector Operations). Let \mathbb{G} be multiplicative cyclic groups of prime p order. Let g be a generator of \mathbb{G} . We define vector operations over \mathbb{G} as follows:

1. For a vector $\vec{b} = (b_1, \dots, b_n) \in \mathbb{Z}_p^n$, define $g^{\vec{b}} := (g^{b_1}, \dots, g^{b_n}) \in \mathbb{G}^n$.
2. For a vector $\vec{b} = (b_1, \dots, b_n) \in \mathbb{Z}_p^n$ and a scalar $c \in \mathbb{Z}_p$, define $(g^{\vec{b}})^c := (g^{b_1 c}, \dots, g^{b_n c}) \in \mathbb{G}^n$.
3. For two vectors $\vec{a} = (a_1, \dots, a_n), \vec{b} = (b_1, \dots, b_n) \in \mathbb{Z}_p^n$, define $g^{\vec{a}} g^{\vec{b}} := (g^{a_1 + b_1}, \dots, g^{a_n + b_n}) \in \mathbb{G}^n$.

Definition 3.2 (Bilinear Product Groups). Let $(p, \mathbb{G}, \mathbb{G}_T, e)$ be bilinear groups of prime order. Let g be a generator of \mathbb{G} . For integers n and m , the bilinear product groups $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_1}, \dots, g^{\vec{b}_m})$ of basis vectors $\vec{b}_1, \dots, \vec{b}_m$ is defined as follows

1. The basis vectors $\vec{b}_1, \dots, \vec{b}_m$ are random vectors such that $\vec{b}_i = (b_{i,1}, \dots, b_{i,n}) \in \mathbb{Z}_p^n$.
2. The bilinear map $e : \mathbb{G}^n \times \mathbb{G}^n \rightarrow \mathbb{G}_T$ is defined as $e(g^{\vec{a}}, g^{\vec{b}}) := \prod_{i=1}^n e(g^{a_i}, g^{b_i}) = e(g, g)^{\vec{a} \cdot \vec{b}}$ where \cdot is the inner product operation.

To guarantee the correctness of cryptographic schemes in bilinear product groups, the orthogonal property of composite order bilinear groups should be implemented in bilinear product groups. The previous research [9, 10, 17, 24] showed that the orthogonal property can be implemented in bilinear product groups. The idea is that the orthogonality between vectors can be defined using the inner-product operation such that $\vec{x} \cdot \vec{y} = 0$ since the bilinear map provides the inner-product operation. Definition 3.3 define the orthogonality in bilinear product groups.

Definition 3.3 (Orthogonality). Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_1}, \dots, g^{\vec{b}_m})$ be bilinear product groups with n, m parameters. Let G_i, G_j be subgroups spanned by $g^{\vec{b}_i}, g^{\vec{b}_j}$, respectively. That is, $G_i = \langle g^{\vec{b}_i} \rangle$ and $G_j = \langle g^{\vec{b}_j} \rangle$. Then the two subgroups G_i and G_j are orthogonal to each other if $e(\vec{A}, \vec{B}) = 1$ for all $\vec{A} \in G_i$ and $\vec{B} \in G_j$.

The main idea of our method that converts HVE schemes from composite order bilinear groups to prime order bilinear groups is that the previous HVE schemes [7, 16, 22] in composite order bilinear groups use the composite 3-party Diffie-Hellman (C3DH) assumption that is not a kind of the subgroup decision (SD) assumption.

The SD assumption is to distinguish whether $h \in \mathbb{G}$ or $h \in \mathbb{G}_1$ where \mathbb{G} is a group and \mathbb{G}_1 is a subgroup of \mathbb{G} [5]. In product groups \mathbb{G}^n , a subgroup G is defined as a vector space spanned by some basis vectors $\vec{b}_1, \dots, \vec{b}_m$ such that $G = \langle g^{\vec{b}_1}, \dots, g^{\vec{b}_m} \rangle$. If a subgroup is constructed from one basis vector, then the SD assumption is related to the DDH assumption. If a subgroup is constructed from k number of basis vectors, then the SD assumption is related to the decisional k -Linear (k -DLIN) assumption [9]. In symmetric bilinear groups of prime order, a subgroup should be constructed from two basis vectors since the DDH assumption is not valid [10, 24]. If a subgroup is constructed from two basis vectors, then cryptographic schemes become complicated and there is no generic conversion method from composite order groups to prime order groups. In asymmetric bilinear groups of prime order, a subgroup can be constructed from one basis vector since the DDH assumption is valid [9, 17]. If a subgroup is constructed from one basis vector, then there is a generic conversion method of Freeman, but it only works in asymmetric bilinear groups.

The C3DH assumption is defined in Assumption 3. The notable properties of the C3DH assumption are that the target value T is always an element of $\mathbb{G}_{p_1 p_2}$ in contrast to the SD assumption, and the subgroup \mathbb{G}_{p_2} plays the role of random blinding. From these properties of the C3DH assumption, it is possible to use just one basis vector to construct a subgroup. Additionally, it is possible to use simple basis vectors for cryptographic schemes since ciphertexts and tokens can use different subgroups that are not orthogonal.

Assumption 3 (Composite 3-party Diffie-Hellman, C3DH). Let $(N, \mathbb{G}, \mathbb{G}_T, e)$ be a description of bilinear groups of composite order $N = p_1 \cdots p_m$ where p_i is a random prime. Let g_{p_i} be a generator of the subgroup \mathbb{G}_{p_i} . The C3DH assumption is stated as follows: given a challenge tuple

$$\vec{D} = ((N, \mathbb{G}, \mathbb{G}_T, e), g_{p_1}, \dots, g_{p_m}, g_{p_1}^a, g_{p_1}^b, g_{p_1}^{ab} R_1, g_{p_1}^{abc} R_2) \text{ and } T,$$

decides whether $T = T_0 = g_{p_1}^c R_3$ or $T = T_1 = g_{p_1}^d R_3$ with random choices of $a, b, c, d \in \mathbb{Z}_{p_1}$ and $R_1, R_2, R_3 \in \mathbb{G}_{p_2}$.

For instance, we select basis vectors $\vec{b}_{1,1} = (1, 0), \vec{b}_{1,2} = (1, a), \vec{b}_2 = (a, -1)$ for the conversion from bilinear groups of composite $N = p_1 p_2$ order. For the conversion from bilinear groups of composite $N = p_1 p_2 p_3$ order, we select basis vectors $\vec{b}_{1,1} = (1, 0, a_1), \vec{b}_{1,2} = (1, a_2, 0), \vec{b}_2 = (a_2, -1, a_1 a_2 - a_3), \vec{b}_3 = (a_1, a_3, -1)$. Although different basis vectors were selected, the assumption for the security proof is the simple one that is independent of basis vectors.

4 Conversion 1: BW-HVE

In this section, we convert the HVE scheme of Boneh and Waters [7] in composite order bilinear groups to an HVE scheme in prime order bilinear groups and prove its selective security under the DBDH and P3DH assumptions.

4.1 Construction

Setup($1^\lambda, \ell$): It first generates the bilinear group \mathbb{G} of prime order p of bit size $\Theta(\lambda)$. It chooses a random value $a \in \mathbb{Z}_p$ and sets basis vectors for bilinear product groups as $\vec{b}_{1,1} = (1, 0)$, $\vec{b}_{1,2} = (1, a)$, $\vec{b}_2 = (a, -1)$. It also sets $\vec{B}_{1,1} = g^{\vec{b}_{1,1}}$, $\vec{B}_{1,2} = g^{\vec{b}_{1,2}}$, $\vec{B}_2 = g^{\vec{b}_2}$. It selects random exponents $v', \{u'_i, h'_i, w'_i\}_{i=1}^\ell, \alpha \in \mathbb{Z}_p, z_v, \{z_{u,i}, z_{h,i}, z_{w,i}\}_{i=1}^\ell \in \mathbb{Z}_p$ and outputs a secret key and a public key as

$$\begin{aligned} SK &= \left(\vec{V}_k = \vec{B}_{1,2}^{v'}, \{ \vec{U}_{k,i} = \vec{B}_{1,2}^{u'_i}, \vec{H}_{k,i} = \vec{B}_{1,2}^{h'_i}, \vec{W}_{k,i} = \vec{B}_{1,2}^{w'_i} \}_{i=1}^\ell, \vec{B}_{1,2}^\alpha \right), \\ PK &= \left(\vec{B}_{1,1}, \vec{B}_{1,2}, \vec{B}_2, \vec{V}_c = \vec{B}_{1,1}^{v'} \vec{B}_2^{z_v}, \{ \vec{U}_{c,i} = \vec{B}_{1,1}^{u'_i} \vec{B}_2^{z_{u,i}}, \vec{H}_{c,i} = \vec{B}_{1,1}^{h'_i} \vec{B}_2^{z_{h,i}}, \vec{W}_{c,i} = \vec{B}_{1,1}^{w'_i} \vec{B}_2^{z_{w,i}} \}_{i=1}^\ell, \right. \\ &\quad \left. \Omega = e(\vec{B}_{1,1}^{v'}, \vec{B}_{1,2})^\alpha \right). \end{aligned}$$

GenToken($\vec{\sigma}, SK, PK$): It takes as input a vector $\vec{\sigma} = (\sigma_1, \dots, \sigma_\ell) \in \Sigma_\ell^*$, the secret key SK , and the public key PK . Let S be the set of indexes that are not wild-card fields in the vector $\vec{\sigma}$. It selects random exponents $\{r_{1,i}, r_{2,i}\}_{i \in S} \in \mathbb{Z}_p$ and outputs a token as

$$TK_{\vec{\sigma}} = \left(\vec{K}_1 = \vec{B}_{1,2}^\alpha \prod_{i \in S} (\vec{U}_{k,i}^{\sigma_i} \vec{H}_{k,i})^{r_{1,i}} \vec{W}_{k,i}^{r_{2,i}}, \{ \vec{K}_{2,i} = \vec{V}_k^{-r_{1,i}}, \vec{K}_{3,i} = \vec{V}_k^{-r_{2,i}} \}_{i \in S} \right).$$

Encrypt(\vec{x}, M, PK): It takes as input a vector $\vec{x} = (x_1, \dots, x_\ell) \in \Sigma^\ell$, a message $M \in \mathcal{M}$, and the public key PK . It first chooses a random exponent $t \in \mathbb{Z}_p$ and random blinding values $z_1, \{z_{2,i}, z_{3,i}\}_{i=1}^\ell \in \mathbb{Z}_p$. Then it outputs a ciphertext as

$$CT = \left(C_0 = \Omega^t M, \vec{C}_1 = \vec{V}_c^t \vec{B}_2^{z_1}, \{ \vec{C}_{2,i} = (\vec{U}_{c,i}^{x_i} \vec{H}_{c,i})^t \vec{B}_2^{z_{2,i}}, \vec{C}_{3,i} = \vec{W}_{c,i}^t \vec{B}_2^{z_{3,i}} \}_{i=1}^\ell \right).$$

Query($CT, TK_{\vec{\sigma}}, PK$): It takes as input a ciphertext CT and a token $TK_{\vec{\sigma}}$ of a vector $\vec{\sigma}$. It first computes

$$M \leftarrow C_0 \cdot \left(e(\vec{C}_1, \vec{K}_1) \cdot \prod_{i \in S} e(\vec{C}_{2,i}, \vec{K}_{2,i}) \cdot e(\vec{C}_{3,i}, \vec{K}_{3,i}) \right)^{-1}.$$

If $M \notin \mathcal{M}$, it outputs \perp indicating that the predicate $f_{\vec{\sigma}}$ is not satisfied. Otherwise, it outputs M indicating that the predicate $f_{\vec{\sigma}}$ is satisfied.

4.2 Correctness

If $f_{\vec{\sigma}}(\vec{x}) = 1$, then the following calculation shows that **Query**($CT, TK_{\vec{\sigma}}, PK$) = M using the orthogonality of basis vectors such that $e(g^{\vec{b}_2}, g^{\vec{b}_{1,2}}) = 1$.

$$\begin{aligned} & e(\vec{C}_1, \vec{K}_1) \cdot \prod_{i \in S} (e(\vec{C}_{2,i}, \vec{K}_{2,i}) \cdot e(\vec{C}_{3,i}, \vec{K}_{3,i})) \\ &= e(\vec{V}_c^t, \vec{B}_{1,2}^\alpha \prod_{i \in S} (\vec{U}_{k,i}^{\sigma_i} \vec{H}_{k,i})^{r_{1,i}} \vec{W}_{k,i}^{r_{2,i}}) \cdot \prod_{i \in S} e((\vec{U}_{c,i}^{x_i} \vec{H}_{c,i})^t, \vec{V}_k^{-r_{1,i}}) \cdot e(\vec{W}_{c,i}^t, \vec{V}_k^{-r_{2,i}}) \\ &= e(\vec{B}_{1,1}^{v't}, \vec{B}_{1,2}^\alpha) \cdot \prod_{i \in S} e(g^{v'}, g^{u'_i(\sigma_i - x_i)})^{t \cdot r_{1,i}} = e(\vec{B}_{1,1}^{v'}, \vec{B}_{1,2})^{\alpha t}. \end{aligned}$$

Otherwise, that is $f_{\vec{\sigma}}(\vec{x}) = 0$, then the probability of **Query**($CT, TK_{\vec{\sigma}}, PK$) $\neq \perp$ is negligible by limiting $|\mathcal{M}|$ to less than $|\mathbb{G}_T|^{1/4}$.

4.3 Security

Theorem 4.1. *The above HVE scheme is selectively secure under the DBDH and P3DH assumptions.*

Proof. The proof of this theorem is easily obtained from the following four Lemmas 4.2, 4.3, 4.4, and 4.5. Before presenting the four lemmas, we first introduce the following three assumptions. The HVE scheme of Boneh and Waters constructed in bilinear groups of composite order $N = p_1 p_2$, and its security was proven under the DBDH, bilinear subgroup decision (BSD), and C3DH assumptions [7]. These assumptions in composite order bilinear groups are converted to the following Assumptions 4-1, 4-2, and 4-3 using our conversion method. \square

Assumption 4-1 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0), \vec{b}_{1,2} = (1, a), \vec{b}_2 = (a, -1)$. The Assumption 4-1 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, (g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,1}})^{c_2}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_3}) \text{ and } T,$$

decides whether $T = T_0 = e(g, g)^{c_1 c_2 c_3}$ or $T = T_1 = e(g, g)^d$ with random choices of $c_1, c_2, c_3, d \in \mathbb{Z}_p$.

Assumption 4-2 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0), \vec{b}_{1,2} = (1, a), \vec{b}_2 = (a, -1)$. The Assumption 4-2 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}) \text{ and } T,$$

decides whether $T = T_0 = e((g^{\vec{b}_{1,1}})^{c_1} (g^{\vec{b}_2})^{c_3}, (g^{\vec{b}_{1,2}})^{c_2})$ or $T = T_1 = e((g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2})$ with random choices of $c_1, c_2, c_3 \in \mathbb{Z}_p$.

Assumption 4-3 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0), \vec{b}_{1,2} = (1, a), \vec{b}_2 = (a, -1)$. The Assumption 4-3 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_1 c_2} (g^{\vec{b}_2})^{z_1}, (g^{\vec{b}_{1,1}})^{c_1 c_2 c_3} (g^{\vec{b}_2})^{z_2}) \text{ and } \vec{T},$$

decides whether $T = T_0 = (g^{\vec{b}_{1,1}})^{c_3} (g^{\vec{b}_2})^{z_3}$ or $T = T_1 = (g^{\vec{b}_{1,1}})^d (g^{\vec{b}_2})^{z_3}$ with random choices of $c_1, c_2, c_3, d \in \mathbb{Z}_p$ and $z_1, z_2, z_3 \in \mathbb{Z}_p$.

Lemma 4.2. *The above HVE scheme is selectively secure under the Assumptions 4-1, 4-2, and 4-3.*

Proof. The proof of this lemma is directly obtained from [7] since the Assumptions 4-1, 4-2, and 4-2 in prime order bilinear groups are correspond to the DBDH, BSD, and C3DH assumptions in composite order bilinear groups. That is, the proof of [7] can be exactly simulated using the vector operations in the Definition 3.1 and the Assumptions 4-1, 4-2, and 4-3. \square

Lemma 4.3. *If the DBDH assumption holds, then the Assumption 4-1 also holds.*

Proof. Suppose there exists an adversary \mathcal{A} that breaks the Assumption 4-1 with a non-negligible advantage. An algorithm \mathcal{B} that solves the DBDH assumption using \mathcal{A} is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), g, g^{c_1}, g^{c_2}, g^{c_3})$ and T where $T = T_0 = e(g, g)^{c_1 c_2 c_3}$ or $T = T_1 = e(g, g)^d$. \mathcal{B} first chooses random values $a \in \mathbb{Z}_p$ and computes

$$\begin{aligned} g^{\vec{b}_{1,1}} &= (g, 1), \quad g^{\vec{b}_{1,2}} = (g, g^a), \quad g^{\vec{b}_2} = (g^a, g^{-1}), \\ (g^{\vec{b}_{1,1}})^{c_1} &= (g^{c_1}, 1), \quad (g^{\vec{b}_{1,1}})^{c_2} = (g^{c_2}, 1), \quad (g^{\vec{b}_{1,1}})^{c_3} = (g^{c_3}, 1), \\ (g^{\vec{b}_{1,2}})^{c_1} &= (g^{c_1}, (g^{c_1})^a), \quad (g^{\vec{b}_{1,2}})^{c_2} = (g^{c_2}, (g^{c_2})^a). \end{aligned}$$

Next, it gives the tuple $D' = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, (g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,1}})^{c_2}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_3})$ and T to \mathcal{A} . Then \mathcal{A} outputs a guess γ' . \mathcal{B} also outputs γ' . If the advantage of \mathcal{A} is ε , then the advantage of \mathcal{B} is greater than ε since the distribution of the challenge tuple to \mathcal{A} is equal to the Assumption 4-1. \square

Lemma 4.4. *The Assumption 4-2 holds for all adversaries.*

Proof. The equation $e((g^{\vec{b}_{1,1}})^{c_1} (g^{\vec{b}_2})^{c_3}, (g^{\vec{b}_{1,2}})^{c_2}) = e((g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2})$ holds by the orthogonality of basis vectors such that $e(g^{\vec{b}_2}, g^{\vec{b}_{1,2}}) = 1$. Therefore, any adversary can not break the Assumption 4-2. \square

Lemma 4.5. *If the P3DH assumption holds, then the Assumption 4-3 also holds.*

Proof. Suppose there exists an adversary \mathcal{A} that breaks the Assumption 4-3 with a non-negligible advantage. An algorithm \mathcal{B} that solves the P3DH assumption using \mathcal{A} is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), (g, f), (g^{c_1}, f^{c_1}), (g^{c_2}, f^{c_2}), (g^{c_1 c_2} f^{z_1}, g^{z_1}), (g^{c_1 c_2 c_3} f^{z_2}, g^{z_2}))$ and T where $T = T_0 = (g^{c_3} f^{z_3}, g^{z_3})$ or $T = T_1 = (g^d f^{z_3}, g^{z_3})$. \mathcal{B} first computes

$$\begin{aligned} g^{\vec{b}_{1,1}} &= (g, 1), g^{\vec{b}_{1,2}} = (g, f), g^{\vec{b}_2} = (f, g^{-1}), \\ (g^{\vec{b}_{1,2}})^{c_1} &= (g^{c_1}, f^{c_1}), (g^{\vec{b}_{1,2}})^{c_2} = (g^{c_2}, f^{c_2}), \\ (g^{\vec{b}_{1,1}})^{c_1 c_2} (g^{\vec{b}_2})^{z_1} &= (g^{c_1 c_2} f^{z_1}, (g^{z_1})^{-1}), (g^{\vec{b}_{1,1}})^{c_1 c_2 c_3} (g^{\vec{b}_2})^{z_2} = (g^{c_1 c_2 c_3} f^{z_2}, (g^{z_2})^{-1}). \end{aligned}$$

Intuitively, it sets $a = \log f$. Next, it gives the tuple $D' = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, (g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,1}})^{c_2}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_1 c_2 c_3})$ and T to \mathcal{A} . Then \mathcal{A} outputs a guess γ' . \mathcal{B} also outputs γ' . If the advantage of \mathcal{A} is ε , then the advantage of \mathcal{B} is greater than ε since the distribution of the challenge tuple to \mathcal{A} is equal to the Assumption 4-3. \square

5 Conversion 2: SW-dHVE

In this section, we convert the delegatable HVE scheme of Shi and Waters [22] to prime order bilinear groups and prove its selective security under the DBDH and P3DH assumptions.

5.1 Construction

Let Σ be a finite set of attributes and let $?, *$ be two special symbol not in Σ . Define $\Sigma_{?,*} = \Sigma \cup \{?, *\}$. The symbol $?$ denotes a delegatable field, i.e., a field where one is allowed to fill in an arbitrary value and perform delegation. The symbol $*$ denotes a wild-card field or “don’t care” field.

Setup($1^\lambda, \ell$): It first generates the bilinear group \mathbb{G} of prime order p of bit size $\Theta(\lambda)$. It chooses random values $a_1, a_2, a_3 \in \mathbb{Z}_p$ and sets basis vectors for bilinear product groups as $\vec{b}_{1,1} = (1, 0, a_1)$, $\vec{b}_{1,2} = (1, a_2, 0)$, $\vec{b}_2 = (a_2, -1, a_1 a_2 - a_3)$, $\vec{b}_3 = (a_1, a_3, -1)$. It also sets

$$\vec{B}_{1,1} = g^{\vec{b}_{1,1}}, \vec{B}_{1,2} = g^{\vec{b}_{1,2}}, \vec{B}_2 = g^{\vec{b}_2}, \vec{B}_3 = g^{\vec{b}_3}.$$

It selects random exponents $v', w'_1, w'_2, \{u'_i, h'_i\}_{i=1}^\ell, \alpha \in \mathbb{Z}_p, z_v, z_{w,1}, z_{w,2}, \{z_{u,i}, z_{h,i}\}_{i=1}^\ell \in \mathbb{Z}_p$ and outputs a secret key and a public key as

$$SK = \left(\vec{V}_k = \vec{B}_{1,2}^{v'}, \vec{W}_{k,1} = \vec{B}_{1,2}^{w'_1}, \vec{W}_{k,2} = \vec{B}_{1,2}^{w'_2}, \{\vec{U}_{k,i} = \vec{B}_{1,2}^{u'_i}, \vec{H}_{k,i} = \vec{B}_{1,2}^{h'_i}\}_{i=1}^\ell, \vec{B}_{1,2}^\alpha \right),$$

$$PK = \left(\vec{B}_{1,1}, \vec{B}_{1,2}, \vec{B}_2, \vec{B}_3, \vec{V}_c = \vec{B}_{1,1}' \vec{B}_2^{z_v}, \vec{W}_{c,1} = \vec{B}_{1,1}' \vec{B}_2^{z_{w,1}}, \vec{W}_{c,2} = \vec{B}_{1,2}' \vec{B}_2^{z_{w,2}}, \right. \\ \left. \{ \vec{U}_{c,i} = \vec{B}_{1,1}' \vec{B}_2^{z_{u,i}}, \vec{H}_{c,i} = \vec{B}_{1,1}' \vec{B}_2^{z_{h,i}} \}_{i=1}^\ell, \Omega = e(\vec{B}_{1,1}', \vec{B}_{1,2})^\alpha \right).$$

GenToken($\vec{\sigma}, SK, PK$): It takes as input an attribute vector $\vec{\sigma} = (\sigma_1, \dots, \sigma_\ell) \in \Sigma_{*,*}^\ell$ and the secret key SK .

1. Let S be the set of indexes that are not delegatable fields and wild-card fields in the vector $\vec{\sigma}$. It first selects random exponents $r_1, r_2, \{r_{3,i}\}_{i \in S} \in \mathbb{Z}_p$ and random blinding values $y_1, y_2, y_3, \{y_{4,i}\}_{i \in S} \in \mathbb{Z}_p$. Then it computes decryption components as

$$\vec{K}_1 = \vec{B}_{1,2}^\alpha \vec{W}_{k,1}^{r_1} \vec{W}_{k,2}^{r_2} \prod_{i \in S} (\vec{U}_{k,i}^{\sigma_i} \vec{H}_{k,i})^{r_{3,i}} \vec{B}_3^{y_1}, \vec{K}_2 = \vec{V}_k^{-r_1} \vec{B}_3^{y_2}, \vec{K}_3 = \vec{V}_k^{-r_2} \vec{B}_3^{y_3}, \\ \{ \vec{K}_{4,i} = \vec{V}_k^{-r_{3,i}} \vec{B}_3^{y_{4,i}} \}_{i \in S}.$$

2. Let $S_?$ be the set of indexes that are delegatable fields. It selects random exponents $\{s_{1,j}, s_{2,j}, \{s_{3,j,i}\}\} \in \mathbb{Z}_p$ and random blinding values $\{y_{1,j,u}, y_{1,j,h}, y_{2,j}, y_{3,j}, \{y_{4,j,i}\}\} \in \mathbb{Z}_p$. Next, it computes delegation components as

$$\forall j \in S_?: \vec{L}_{1,j,u} = \vec{U}_{k,i}^{s_{3,j,j}} \vec{B}_3^{y_{1,j,u}}, \vec{L}_{1,j,h} = \vec{W}_{k,1}^{s_{1,j}} \vec{W}_{k,2}^{s_{2,j}} \prod_{i \in S} (\vec{U}_{k,i}^{\sigma_i} \vec{H}_{k,i})^{s_{3,j,i}} \vec{H}_{k,j}^{s_{3,j,j}} \vec{B}_3^{y_{1,j,h}}, \\ \vec{L}_{2,j} = \vec{V}_k^{-s_{1,j}} \vec{B}_3^{y_{2,j}}, \vec{L}_{3,j} = \vec{V}_k^{-s_{2,j}} \vec{B}_3^{y_{3,j}}, \{ \vec{L}_{4,j,i} = \vec{V}_k^{-s_{3,j,i}} \vec{B}_3^{y_{4,j,i}} \}_{i \in S \cup \{j\}}.$$

3. Finally, it outputs a token as

$$TK_{\vec{\sigma}} = \left(\vec{K}_1, \vec{K}_2, \vec{K}_3, \{ \vec{K}_{4,i} \}_{i \in S}, \{ \vec{L}_{1,j,u}, \vec{L}_{1,j,h}, \vec{L}_{2,j}, \vec{L}_{3,j}, \{ \vec{L}_{4,j,i} \}_{i \in S \cup \{j\}} \}_{j \in S_?} \right).$$

Delegate($\vec{\sigma}', TK_{\vec{\sigma}}, PK$): It takes as input an attribute vector $\vec{\sigma}' = (\sigma_1, \dots, \sigma_\ell) \in \Sigma_{*,*}^\ell$ and a token $TK_{\vec{\sigma}}$. Without loss of generality, we assume that σ' fixes only one delegatable field of σ . It is clear that we can perform delegation on multiple fields if we have an algorithm to perform delegation on one field. Suppose σ' fixes the k -th index of σ .

1. If the k -th index of σ' is set to $*$, that is, a wild-card field, then it can perform delegation by simply removing the delegation components that correspond to k -th index.
2. Otherwise, that is, if the k -th index of σ' is set to some value in Σ , then it perform delegation as follows:

- (a) Let S be the set of indexes that are not delegatable fields and wild-card fields in the vector $\vec{\sigma}'$. Note that $k \in S$. It selects random exponents $\mu, y_1, y_2, y_3, \{y_{4,i}\}_{i \in S} \in \mathbb{Z}_p$ and updates the token as

$$\vec{K}'_1 = \vec{K}_1 (\vec{L}_{1,k,u}^{\sigma_k} \vec{L}_{1,k,h})^\mu \vec{B}_3^{y_1}, \vec{K}'_2 = \vec{K}_2 \vec{L}_{2,k}^{y_2}, \vec{K}'_3 = \vec{K}_3 \vec{L}_{3,k}^{y_3}, \\ \vec{K}'_{4,k} = \vec{L}_{4,k}^\mu \vec{B}_3^{y_{4,k}}, \{ \vec{K}'_{4,i} = \vec{K}_{4,i} \vec{L}_{4,k,i}^\mu \vec{B}_3^{y_{4,i}} \}_{i \in S \setminus \{k\}}.$$

- (b) Let $S_?$ be the set of indexes that are delegatable fields in the vector $\vec{\sigma}'$. It selects random exponents $\{\tau_j, y_{1,j,u}, y_{1,j,h}, y_{2,j}, y_{3,j}, \{y_{4,j,i}\}_{i \in S \cup \{j\}}\}_{j \in S_?} \in \mathbb{Z}_p$ and re-randomize the delegation components of the token as

$$\forall j \in S_?: \vec{L}'_{1,j,u} = \vec{L}_{1,j,u}^\mu \vec{B}_3^{y_{1,j,u}}, \vec{L}'_{1,j,h} = \vec{L}_{1,j,h}^\mu (\vec{L}_{1,k,u}^{\sigma_k} \vec{L}_{1,k,h})^{\tau_j} \vec{B}_3^{y_{1,j,h}}, \\ \vec{L}'_{2,j} = \vec{L}_{2,j}^\mu \vec{L}_{2,j}^{\tau_j} \vec{B}_3^{y_{2,j}}, \vec{L}'_{3,j} = \vec{L}_{3,j}^\mu \vec{L}_{3,j}^{\tau_j} \vec{B}_3^{y_{3,j}}, \\ \vec{L}'_{4,j,j} = \vec{L}_{4,j,j}^\mu \vec{B}_3^{y_{4,j,j}}, \vec{L}'_{4,j,k} = \vec{L}_{4,j,k}^{\tau_j} \vec{B}_3^{y_{4,j,k}}, \{ \vec{L}'_{4,j,i} = \vec{L}_{4,j,i}^\mu \vec{L}_{4,j,k}^{\tau_j} \vec{B}_3^{y_{4,j,i}} \}_{i \in S \setminus \{k\}}.$$

(c) Finally, it outputs a token as

$$TK_{\vec{\sigma}} = \left(\vec{K}'_1, \vec{K}'_2, \vec{K}'_3, \{\vec{K}'_{4,i}\}_{i \in S}, \{\vec{L}'_{1,j,h}, \vec{L}'_{1,j,u}, \vec{L}'_{2,j}, \vec{L}'_{3,j}, \{\vec{L}'_{4,j,i}\}_{i \in S \cup \{j\}}\}_{j \in S_?} \right).$$

Encrypt(\vec{x}, M, PK): It takes as input an attribute vector $\vec{x} = (x_1, \dots, x_\ell) \in \Sigma^\ell$, a message $M \in \mathcal{M} \subseteq \mathbb{G}_T$, and the public key PK . It first chooses a random exponent $t \in \mathbb{Z}_p$ and random blinding values $z_1, z_2, z_3, \{z_{4,i}\}_{i=1}^\ell \in \mathbb{Z}_p$. Then it outputs a ciphertext as

$$CT = \left(C_0 = \Omega^t M, \vec{C}_1 = \vec{V}_c^t \vec{B}_2^{z_1}, \vec{C}_2 = \vec{W}_{c,1}^t \vec{B}_2^{z_2}, \vec{C}_3 = \vec{W}_{c,2}^t \vec{B}_2^{z_3}, \{\vec{C}_{4,i} = (\vec{U}_{c,i}^{x_i} \vec{H}_{c,i})^t \vec{B}_2^{z_{4,i}}\}_{i=1}^\ell \right).$$

Query($CT, TK_{\vec{\sigma}}, PK$): It takes as input a ciphertext CT and a token $TK_{\vec{\sigma}}$ of a vector $\vec{\sigma}$. It first computes

$$M \leftarrow C_0 \cdot \left(e(\vec{C}_1, \vec{K}_1) \cdot e(\vec{C}_2, \vec{K}_2) \cdot e(\vec{C}_3, \vec{K}_3) \cdot \prod_{i \in S} e(\vec{C}_{4,i}, \vec{K}_{4,i}) \right)^{-1}.$$

If $M \notin \mathcal{M}$, it outputs \perp indicating that the predicate $f_{\vec{\sigma}}$ is not satisfied. Otherwise, it outputs M indicating that the predicate $f_{\vec{\sigma}}$ is satisfied.

5.2 Correctness

If $f_{\vec{\sigma}}(\vec{x}) = 1$, then the following calculation shows that **Query**($CT, TK_{\vec{\sigma}}, PK$) = M by the orthogonality of basis vectors such that $e(g^{\vec{b}_{1,1}}, g^{\vec{b}_3}) = 1, e(g^{\vec{b}_{1,2}}, g^{\vec{b}_2}) = 1, e(g^{\vec{b}_2}, g^{\vec{b}_3}) = 1$.

$$\begin{aligned} & e(\vec{C}_1, \vec{K}_1) \cdot e(\vec{C}_2, \vec{K}_2) \cdot e(\vec{C}_3, \vec{K}_3) \cdot \prod_{i \in S} e(\vec{C}_{4,i}, \vec{K}_{4,i}) \\ &= e((\vec{V}_c)^t, \vec{B}_{1,2}^\alpha \vec{W}_{k,1}^{r_1} \vec{W}_{k,2}^{r_2} \prod_{i \in S} (\vec{U}_{c,i}^{\sigma_i} \vec{H}_{c,i})^{r_{3,i}}) \cdot e(\vec{W}_{c,1}^t, \vec{V}_k^{-r_1}) \cdot e(\vec{W}_{c,2}^t, \vec{V}_k^{-r_2}) \cdot \prod_{i \in S} e((\vec{U}_{c,i}^{x_i} \vec{H}_{c,i})^t, \vec{V}_k^{-r_{3,i}}) \\ &= e(\vec{B}_{1,1}^{v'_t}, \vec{B}_{1,2}^\alpha) \cdot \prod_{i \in S} e(g^{v'}, g^{u'_i(\sigma_i - x_i)})^{tr_{3,i}} = e(\vec{B}_{1,1}^{v'}, \vec{B}_{1,2}^\alpha)^{\alpha t}. \end{aligned}$$

Otherwise, that is $f_{\vec{\sigma}}(\vec{x}) = 0$, the probability of **Query**($CT, TK_{\vec{\sigma}}, PK$) $\neq \perp$ is negligible by limiting $|\mathcal{M}|$ to less than $|\mathbb{G}_T|^{1/4}$.

5.3 Security

Theorem 5.1. *The above dHVE scheme is selectively secure under the DBDH and P3DH assumptions.*

Proof. The proof of this theorem is easily obtained from the following five Lemmas 5.2, 5.3, 5.4, 5.5 and 5.6. Before presenting the five lemmas, we first introduce the following four assumptions. The HVE scheme of Shi and Waters constructed in bilinear groups of composite order $N = p_1 p_2 p_3$, and its security was proven under the DBDH, BSD, and C3DH assumptions [22]. In composite order bilinear groups, the C3DH assumption imply the l -C3DH assumption that was introduced in [22]. However, this implication is not valid in prime order bilinear groups since the basis vectors for ciphertexts and tokens are different. Thus the C3DH assumption for ciphertexts and the C3DH assumption for tokens should be treated as differently. These assumptions in composite order bilinear groups are converted to the following Assumptions 5-1, 5-2, 5-3, and 5-4 using our conversion method. \square

Assumption 5-1 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0, a_1)$, $\vec{b}_{1,2} = (1, a_2, 0)$, $\vec{b}_2 = (a_2, -1, a_1 a_2 - a_3)$, $\vec{b}_3 = (a_1, a_3, -1)$. The Assumption 5-1 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,1}})^{c_2}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_3}) \text{ and } T,$$

decides whether $T = T_0 = e(g, g)^{c_1 c_2 c_3}$ or $T = T_1 = e(g, g)^d$ with random choices of $c_1, c_2, c_3, d \in \mathbb{Z}_p$.

Assumption 5-2 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0, a_1)$, $\vec{b}_{1,2} = (1, a_2, 0)$, $\vec{b}_2 = (a_2, -1, a_1 a_2 - a_3)$, $\vec{b}_3 = (a_1, a_3, -1)$. The Assumption 5-2 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}) \text{ and } T,$$

decides whether $T = T_0 = e((g^{\vec{b}_{1,1}})^{c_1} (g^{\vec{b}_2})^{c_3}, (g^{\vec{b}_{1,2}})^{c_2} (g^{\vec{b}_3})^{c_4})$ or $T = T_1 = e((g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2})$ with random choices of $c_1, c_2, c_3, c_4 \in \mathbb{Z}_p$.

Assumption 5-3 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0, a_1)$, $\vec{b}_{1,2} = (1, a_2, 0)$, $\vec{b}_2 = (a_2, -1, a_1 a_2 - a_3)$, $\vec{b}_3 = (a_1, a_3, -1)$. The Assumption 5-3 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_1 c_2} (g^{\vec{b}_2})^{z_1}, (g^{\vec{b}_{1,1}})^{c_1 c_2 c_3} (g^{\vec{b}_2})^{z_2}) \text{ and } T,$$

decides whether $T = T_0 = (g^{\vec{b}_{1,1}})^{c_3} (g^{\vec{b}_2})^{z_3}$ or $T = T_1 = (g^{\vec{b}_{1,1}})^d (g^{\vec{b}_2})^{z_3}$ with random choices of $c_1, c_2, c_3, d \in \mathbb{Z}_p$, and $z_1, z_2, z_3 \in \mathbb{Z}_p$.

Assumption 5-4 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0, a_1)$, $\vec{b}_{1,2} = (1, a_2, 0)$, $\vec{b}_2 = (a_2, -1, a_1 a_2 - a_3)$, $\vec{b}_3 = (a_1, a_3, -1)$. The Assumption 5-4 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,1}})^{c_2}, (g^{\vec{b}_{1,2}})^{c_1 c_2} (g^{\vec{b}_3})^{z_1}, (g^{\vec{b}_{1,2}})^{c_1 c_2 c_3} (g^{\vec{b}_3})^{z_2}) \text{ and } T,$$

decides whether $T = T_0 = (g^{\vec{b}_{1,2}})^{c_3} (g^{\vec{b}_3})^{z_3}$ or $T = T_1 = (g^{\vec{b}_{1,2}})^d (g^{\vec{b}_3})^{z_3}$ with random choices of $c_1, c_2, c_3, d \in \mathbb{Z}_p$, and $z_1, z_2, z_3 \in \mathbb{Z}_p$.

Lemma 5.2. *The above dHVE scheme is selectively secure under the Assumptions 5-1, 5-2, 5-3, and 5-4.*

Proof. The proof of this lemma is directly obtained from [22] since the Assumptions 5-1, 5-2, 5-3, and 5-4 in prime order bilinear groups are correspond to the DBDH, BSD, C3DH (for ciphertexts), and C3DH (for tokens) assumptions in composite order bilinear groups. \square

Lemma 5.3. *If the DBDH assumption holds, then the Assumption 5-1 also holds.*

Proof. Suppose there exists an adversary \mathcal{A} that breaks the Assumption 5-1 with a non-negligible advantage. An algorithm \mathcal{B} that solves the DBDH assumption using \mathcal{A} is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), g, g^{c_1}, g^{c_2}, g^{c_3})$ and T where $T = T_0 = e(g, g)^{c_1 c_2 c_3}$ or $T = T_1 = e(g, g)^d$. \mathcal{B} first chooses random values $a_1, a_2, a_3 \in \mathbb{Z}_p$ and sets

$$\begin{aligned} g^{\vec{b}_{1,1}} &= (g, 1, g^{a_1}), g^{\vec{b}_{1,2}} = (g, g^{a_2}, 1), g^{\vec{b}_2} = (g^{a_2}, g^{-1}, g^{a_1 a_2 - a_3}), g^{\vec{b}_3} = (g^{a_1}, g^{a_3}, g^{-1}), \\ (g^{\vec{b}_{1,1}})^{c_1} &= (g^{c_1}, 1, (g^{c_1})^{a_1}), (g^{\vec{b}_{1,1}})^{c_2} = (g^{c_2}, 1, (g^{c_2})^{a_1}), (g^{\vec{b}_{1,1}})^{c_3} = (g^{c_3}, 1), \\ (g^{\vec{b}_{1,2}})^{c_1} &= (g^{c_1}, (g^{c_1})^{a_2}, 1), (g^{\vec{b}_{1,2}})^{c_2} = (g^{c_2}, (g^{c_2})^{a_2}, 1). \end{aligned}$$

Next, it gives the tuple $D' = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, (g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,1}})^{c_2}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_3})$ and T to \mathcal{A} . Then \mathcal{A} outputs a guess γ' . \mathcal{B} also outputs γ' . If the advantage of \mathcal{A} is ε , then the advantage of \mathcal{B} is greater than ε since the distribution of the challenge tuple to \mathcal{A} is equal to the Assumption 5-1. \square

Lemma 5.4. *The Assumption 5-2 holds for all adversaries.*

Proof. The equation $e((g^{\vec{b}_{1,1}})^{c_1} (g^{\vec{b}_2})^{c_3}, (g^{\vec{b}_{1,2}})^{c_2} (g^{\vec{b}_3})^{c_4}) = e((g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2})$ holds by the orthogonality of basis vectors such that $e(g^{\vec{b}_{1,1}}, g^{\vec{b}_3}) = 1, e(g^{\vec{b}_2}, g^{\vec{b}_{1,2}}) = 1, e(g^{\vec{b}_2}, g^{\vec{b}_3}) = 1$. Therefore, any adversary can not break the Assumption 5-2. \square

Lemma 5.5. *If the P3DH assumption holds, then the Assumption 5-3 also holds.*

Proof. Suppose there exists an adversary \mathcal{A} that breaks the Assumption 5-3 with a non-negligible advantage. An algorithm \mathcal{B} that solves the P3DH assumption using \mathcal{A} is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), (g, f), (g^{c_1}, f^{c_1}), (g^{c_2}, f^{c_2}), (g^{c_1 c_2} f^{z_1}, g^{z_1}), (g^{c_1 c_2 c_3} f^{z_2}, g^{z_2}))$ and $T = T_\gamma = (T_{\gamma,1}, T_{\gamma,2})$ where $T = T_0 = (g^{c_3} f^{z_3}, g^{z_3})$ or $T = T_1 = (g^d f^{z_3}, g^{z_3})$. \mathcal{B} first chooses random values $a_1, a_3 \in \mathbb{Z}_p$ and sets

$$\begin{aligned} g^{\vec{b}_{1,1}} &= (g, 1, g^{a_1}), g^{\vec{b}_{1,2}} = (g, f, 1), g^{\vec{b}_2} = (f, g^{-1}, f^{a_1} g^{-a_3}), g^{\vec{b}_3} = (g^{a_1}, g^{a_3}, g^{-1}), \\ (g^{\vec{b}_{1,2}})^{c_1} &= (g^{c_1}, f^{c_1}, 1), (g^{\vec{b}_{1,2}})^{c_2} = (g^{c_2}, f^{c_2}, 1), \\ (g^{\vec{b}_{1,1}})^{c_1 c_2} (g^{\vec{b}_2})^{z_1} &= (g^{c_1 c_2} f^{z_1}, (g^{z_1})^{-1}, (g^{c_1 c_2} f^{z_1})^{a_1} (g^{z_1})^{-a_3}), \\ (g^{\vec{b}_{1,1}})^{c_1 c_2 c_3} (g^{\vec{b}_2})^{z_2} &= (g^{c_1 c_2 c_3} f^{z_2}, (g^{z_2})^{-1}, (g^{c_1 c_2 c_3} f^{z_2})^{a_1} (g^{z_2})^{-a_3}), \\ T' &= (T_{\gamma,1}, T_{\gamma,2}, (T_{\gamma,1})^{a_1} (T_{\gamma,2})^{-a_3}). \end{aligned}$$

Intuitively, it sets $a_2 = \log f$. Next, it gives the tuple $D' = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_1 c_2} (g^{\vec{b}_2})^{z_1}, (g^{\vec{b}_{1,1}})^{c_1 c_2 c_3} (g^{\vec{b}_2})^{z_2})$ and T' to \mathcal{A} . Then \mathcal{A} outputs a guess γ' . \mathcal{B} also outputs γ' . If the advantage of \mathcal{A} is ε , then the advantage of \mathcal{B} is greater than ε since the distribution of the challenge tuple to \mathcal{A} is equal to the Assumption 5-3. \square

Lemma 5.6. *If the P3DH assumption holds, then the Assumption 5-4 also holds.*

Proof. Suppose there exists an adversary \mathcal{A} that breaks the Assumption 5-4 with a non-negligible advantage. An algorithm \mathcal{B} that solves the P3DH assumption using \mathcal{A} is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), (g, f), (g^{c_1}, f^{c_1}), (g^{c_2}, f^{c_2}), (g^{c_1 c_2} f^{z_1}, g^{z_1}), (g^{c_1 c_2 c_3} f^{z_2}, g^{z_2}))$ and $T = T_\gamma = (T_{\gamma,1}, T_{\gamma,2})$ where $T_0 = (g^{c_3} f^{z_3}, g^{z_3})$ or $T_1 = (g^d f^{z_3}, g^{z_3})$. \mathcal{B} first chooses random values $a_2, a_3 \in \mathbb{Z}_p$ and sets

$$\begin{aligned} g^{\vec{b}_{1,1}} &= (g, 1, f), g^{\vec{b}_{1,2}} = (g, g^{a_2}, 1), g^{\vec{b}_2} = (g^{a_2}, g^{-1}, g^{a_3}), g^{\vec{b}_3} = (f, f^{a_2} g^{-a_3}, g^{-1}), \\ (g^{\vec{b}_{1,1}})^{c_1} &= (g^{c_1}, 1, f^{c_1}), (g^{\vec{b}_{1,1}})^{c_2} = (g^{c_2}, 1, f^{c_2}), \\ (g^{\vec{b}_{1,2}})^{c_1 c_2} (g^{\vec{b}_3})^{z_1} &= (g^{c_1 c_2} f^{z_1}, (g^{c_1 c_2} f^{z_1})^{a_2} (g^{z_1})^{-a_3}, (g^{z_1})^{-1}), \\ (g^{\vec{b}_{1,2}})^{c_1 c_2 c_3} (g^{\vec{b}_3})^{z_2} &= (g^{c_1 c_2 c_3} f^{z_2}, (g^{c_1 c_2 c_3} f^{z_2})^{a_2} (g^{z_2})^{-a_3}, (g^{z_2})^{-1}), \\ T' &= (T_{\gamma,1}, (T_{\gamma,1})^{a_2} (T_{\gamma,2})^{-a_3}, (T_{\gamma,2})^{-1}). \end{aligned}$$

Intuitively, it sets $a'_1 = \log f, a'_2 = a_2, a'_3 = a_1 a_2 - a_3$ where a'_1, a'_2, a'_3 are elements of basis vectors for the Assumption 5-4. Next, it gives the tuple $D' = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,1}})^{c_2}, (g^{\vec{b}_{1,2}})^{c_1 c_2}, (g^{\vec{b}_3})^{z_1}, (g^{\vec{b}_{1,2}})^{c_1 c_2 c_3} (g^{\vec{b}_3})^{z_2})$ and T' to \mathcal{A} . Then \mathcal{A} outputs a guess γ' . \mathcal{B} also outputs γ' . If the advantage of \mathcal{A} is ε , then the advantage of \mathcal{B} is greater than ε since the distribution of the challenge tuple to \mathcal{A} is equal to the Assumption 5-4. \square

6 Conversion 3: LL-HVE

In this section, we convert the HVE scheme of Lee and Lee [16] to prime order bilinear groups and prove its selective security under the DBDH and P3DH assumptions.

6.1 Construction

Setup($1^\lambda, \ell$): It generates the bilinear group \mathbb{G} of prime order p of bit size $\Theta(\lambda)$. It chooses random values $a_1, a_2, a_3 \in \mathbb{Z}_p$ and sets basis vectors for bilinear product groups as $\vec{b}_{1,1} = (1, 0, a_1)$, $\vec{b}_{1,2} = (1, a_2, 0)$, $\vec{b}_2 = (a_2, -1, a_1 a_2 - a_3)$, $\vec{b}_3 = (a_1, a_3, -1)$. It also sets

$$\vec{B}_{1,1} = g^{\vec{b}_{1,1}}, \vec{B}_{1,2} = g^{\vec{b}_{1,2}}, \vec{B}_2 = g^{\vec{b}_2}, \vec{B}_3 = g^{\vec{b}_3}.$$

It selects random exponents $v', w'_1, w'_2, \{u'_i, h_i\}_{i=1}^\ell, \alpha \in \mathbb{Z}_p, z_v, z_{w,1}, z_{w,2}, \{z_{u,i}, z_{h,i}\}_{i=1}^\ell \in \mathbb{Z}_p$ and outputs a secret key and a public key as

$$\begin{aligned} SK &= \left(V_k = \vec{B}_{1,2}^{v'}, W_{k,1} = \vec{B}_{1,2}^{w'_1}, W_{k,2} = \vec{B}_{1,2}^{w'_2}, \{U_{k,i} = \vec{B}_{1,2}^{u'_i}, H_{k,i} = \vec{B}_{1,2}^{h_i}\}_{i=1}^\ell, \vec{B}_{1,2}^\alpha \right), \\ PK &= \left(\vec{B}_{1,1}, \vec{B}_{1,2}, \vec{B}_2, \vec{B}_3, \vec{V}_c = \vec{B}_{1,1}^{v'} \vec{B}_2^{z_v}, \vec{W}_{c,1} = \vec{B}_{1,1}^{w'_1} \vec{B}_2^{z_{w,1}}, \vec{W}_{c,2} = \vec{B}_{1,1}^{w'_2} \vec{B}_2^{z_{w,2}}, \right. \\ &\quad \left. \{\vec{U}_{c,i} = \vec{B}_{1,1}^{u'_i} \vec{B}_2^{z_{u,i}}, \vec{H}_{c,i} = \vec{B}_{1,1}^{h_i} \vec{B}_2^{z_{h,i}}\}_{i=1}^\ell, \Omega = e(\vec{B}_{1,1}, \vec{B}_{1,2})^\alpha \right). \end{aligned}$$

GenToken($\vec{\sigma}, SK, PK$): It takes as input a vector $\vec{\sigma} = (\sigma_1, \dots, \sigma_\ell) \in \Sigma_*^\ell$ and the secret key SK . Let S be the set of indexes that are not wild-card fields in the vector $\vec{\sigma}$. It selects random exponents $r_1, r_2, r_3 \in \mathbb{Z}_p$ and random blinding values $y_1, y_2, y_3, y_4 \in \mathbb{Z}_p$. Next it outputs a token as

$$TK_{\vec{\sigma}} = \left(\vec{K}_1 = \vec{B}_{1,2}^\alpha \vec{W}_{k,1}^{r_1} \vec{W}_{k,2}^{r_2} \prod_{i \in S} (\vec{U}_{k,i}^{\sigma_i} \vec{H}_{k,i})^{r_3} \vec{B}_3^{y_1}, \vec{K}_2 = \vec{V}_c^{-r_1} \vec{B}_3^{y_2}, \vec{K}_3 = \vec{V}_c^{-r_2} \vec{B}_3^{y_3}, \vec{K}_4 = \vec{V}_c^{-r_3} \vec{B}_3^{y_4} \right).$$

Encrypt(\vec{x}, M, PK): It takes as input a vector $\vec{x} = (x_1, \dots, x_\ell) \in \Sigma^l$, a message $M \in \mathcal{M}$, and the public key PK . It first chooses a random exponent $t \in \mathbb{Z}_p$ and random blinding values $z_1, z_2, z_3, \{z_{4,i}\}_{i=1}^\ell \in \mathbb{Z}_p$. Then it outputs a ciphertext as

$$CT = \left(C_0 = \Omega^t M, \vec{C}_1 = \vec{V}_c^t \vec{B}_2^{z_1}, \vec{C}_2 = \vec{W}_{c,1}^t \vec{B}_2^{z_2}, \vec{C}_3 = \vec{W}_{c,2}^t \vec{B}_2^{z_3}, \{\vec{C}_{4,i} = (\vec{U}_{c,i}^{x_i} \vec{H}_{c,i})^t \vec{B}_2^{z_{4,i}}\}_{i=1}^\ell \right).$$

Query($CT, TK_{\vec{\sigma}}, PK$): It takes as input a ciphertext CT and a token $TK_{\vec{\sigma}}$ of a vector $\vec{\sigma}$. It first computes

$$M \leftarrow C_0 \cdot \left(e(\vec{C}_1, \vec{K}_1) \cdot e(\vec{C}_2, \vec{K}_2) \cdot e(\vec{C}_3, \vec{K}_3) \cdot e\left(\prod_{i \in S} \vec{C}_{4,i}, \vec{K}_4\right) \right)^{-1}.$$

If $M \notin \mathcal{M}$, it outputs \perp indicating that the predicate $f_{\vec{\sigma}}$ is not satisfied. Otherwise, it outputs M indicating that the predicate $f_{\vec{\sigma}}$ is satisfied.

6.2 Correctness

If $f_{\vec{\sigma}}(\vec{x}) = 1$, then the following calculation shows that $\mathbf{Query}(CT, TK_{\vec{\sigma}}, PK) = M$ by the orthogonality of basis vectors such that $e(g^{\vec{b}_{1,1}}, g^{\vec{b}_3}) = 1, e(g^{\vec{b}_{1,2}}, g^{\vec{b}_2}) = 1, e(g^{\vec{b}_2}, g^{\vec{b}_3}) = 1$.

$$\begin{aligned} & e(\vec{C}_1, \vec{K}_1) \cdot e(\vec{C}_2, \vec{K}_2) \cdot e(\vec{C}_3, \vec{K}_3) \cdot e\left(\prod_{i \in S} \vec{C}_{4,i}, \vec{K}_4\right) \\ &= e(\vec{V}_c^t, \vec{B}_{1,2}^\alpha \vec{W}_{k,1}^{r_1} \vec{W}_{k,2}^{r_2} \prod_{i \in S} (\vec{U}_{k,i}^{\sigma_i} \vec{H}_{k,i})^{r_3}) \cdot e(\vec{W}_{c,1}^t, \vec{V}_k^{-r_1}) \cdot e(\vec{W}_{c,2}^t, \vec{V}_k^{-r_2}) \cdot e\left(\prod_{i \in S} (\vec{U}_{c,i}^{x_i} \vec{H}_{c,i})^t, \vec{V}_k^{-r_3}\right) \\ &= e(\vec{B}_{1,1}^{v'}, \vec{B}_{1,2}^\alpha) \cdot e(g^{v'}, \prod_{i \in S} g^{u'_i(\sigma_i - x_i)})^{tr_3} = e(\vec{B}_{1,1}^{v'}, \vec{B}_{1,2}^\alpha)^{\alpha t}. \end{aligned}$$

Otherwise, that is $f_{\vec{\sigma}}(\vec{x}) = 0$, the probability of $\mathbf{Query}(CT, TK_{\vec{\sigma}}, PK) \neq \perp$ is negligible by limiting $|\mathcal{M}|$ to less than $|\mathbb{G}_T|^{1/4}$.

6.3 Security

Theorem 6.1. *The above HVE scheme is selectively secure under the DBDH and P3DH assumptions.*

Proof. The proof of this theorem is easily obtained from the following five Lemmas 6.2, 6.3, 6.4, 6.5 and 6.6. Before presenting the five lemmas, we first introduce the following four assumptions. The HVE scheme of Lee and Lee constructed in bilinear groups of composite order $N = p_1 p_2 p_3$, and its security was proven under the DBDH, BSD, and C3DH assumptions [22]. In composite order bilinear groups, the C3DH assumption imply the C2DH assumption that was introduced in [16]. However, this implication is not valid in prime order bilinear groups since the basis vectors for ciphertexts and tokens are different. Thus the C3DH assumption for ciphertexts and the C2DH assumption for tokens should be treated as differently. These assumptions in composite order bilinear groups are converted to the following Assumptions 6-1, 6-2, 6-3, and 6-4 using our conversion method. \square

Assumption 6-1 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0, a_1), \vec{b}_{1,2} = (1, a_2, 0), \vec{b}_2 = (a_2, -1, a_1 a_2 - a_3), \vec{b}_3 = (a_1, a_3, -1)$. The Assumption 6-1 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,1}})^{c_2}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_3}) \text{ and } T,$$

decides whether $T = T_0 = e(g, g)^{c_1 c_2 c_3}$ or $T = T_1 = e(g, g)^d$ with random choices of $c_1, c_2, c_3, d \in \mathbb{Z}_p$.

Assumption 6-2 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0, a_1), \vec{b}_{1,2} = (1, a_2, 0), \vec{b}_2 = (a_2, -1, a_1 a_2 - a_3), \vec{b}_3 = (a_1, a_3, -1)$. The Assumption 6-2 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}) \text{ and } T,$$

decides whether $T = T_0 = e((g^{\vec{b}_{1,1}})^{c_1} (g^{\vec{b}_2})^{c_3}, (g^{\vec{b}_{1,2}})^{c_2} (g^{\vec{b}_3})^{c_4})$ or $T = T_1 = e((g^{\vec{b}_{1,1}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2})$ with random choices of $c_1, c_2, c_3, c_4 \in \mathbb{Z}_p$.

Assumption 6-3 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0, a_1), \vec{b}_{1,2} = (1, a_2, 0), \vec{b}_2 = (a_2, -1, a_1 a_2 - a_3), \vec{b}_3 = (a_1, a_3, -1)$. The Assumption 6-3 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,2}})^{c_1}, (g^{\vec{b}_{1,2}})^{c_2}, (g^{\vec{b}_{1,1}})^{c_1 c_2} (g^{\vec{b}_2})^{z_1}, (g^{\vec{b}_{1,1}})^{c_1 c_2 c_3} (g^{\vec{b}_2})^{z_2}) \text{ and } T,$$

decides whether $T = T_0 = (g^{\vec{b}_{1,1}})^{c_3} (g^{\vec{b}_2})^{z_3}$ or $T = T_1 = (g^{\vec{b}_{1,1}})^d (g^{\vec{b}_2})^{z_3}$ with random choices of $c_1, c_2, c_3, d \in \mathbb{Z}_p$ and $z_1, z_2, z_3 \in \mathbb{Z}_p$.

Assumption 6-4 Let $((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3})$ be the bilinear product group of basis vectors $\vec{b}_{1,1} = (1, 0, a_1), \vec{b}_{1,2} = (1, a_2, 0), \vec{b}_2 = (a_2, -1, a_1 a_2 - a_3), \vec{b}_3 = (a_1, a_3, -1)$. The Assumption 6-4 is stated as follows: given a challenge tuple

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,2}})^{c_1} (g^{\vec{b}_3})^{z_1}, (g^{\vec{b}_{1,2}})^{c_2} (g^{\vec{b}_3})^{z_2}) \text{ and } T,$$

decides whether $T = T_0 = (g^{\vec{b}_{1,2}})^{c_1 c_2} (g^{\vec{b}_3})^{z_3}$ or $T = T_1 = (g^{\vec{b}_{1,2}})^d (g^{\vec{b}_3})^{z_3}$ with random choices of $c_1, c_2, d \in \mathbb{Z}_p$ and $z_1, z_2, z_3 \in \mathbb{Z}_p$.

Lemma 6.2. *The above HVE scheme is selectively secure under the Assumptions 6-1, 6-2, 6-3, and 6-4.*

Proof. The proof of this lemma is directly obtained from [16] since the Assumptions 6-1, 6-2, 6-3, and 6-4 in prime order bilinear groups corresponds to the DBDH, BSD, C3DH, and C2DH assumptions in composite order bilinear groups. \square

Lemma 6.3. *If the DBDH assumption holds, then the Assumption 6-1 also holds.*

Lemma 6.4. *The Assumption 6-2 holds for all adversaries.*

Lemma 6.5. *If the P3DH assumption holds, then the Assumption 6-3 also holds.*

The Assumptions 6-1, 6-2, and 6-3 are the same as the Assumptions 5-1, 5-2, and 5-3. Thus we omits the proofs of Lemmas 6.3, 6.4, 6.5.

Lemma 6.6. *If the P3DH assumption holds, then the Assumption 6-4 also holds.*

Proof. Suppose there exists an adversary \mathcal{A} that breaks the Assumption 6-4 with a non-negligible advantage. An algorithm \mathcal{B} that solves the P3DH assumption using \mathcal{A} is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), (g, f), (g^{c_1}, f^{c_1}), (g^{c_2}, f^{c_2}), (g^{c_1 c_2} f^{z_1}, g^{z_1}), (g^{c_3} f^{z_2}, g^{z_2}))$ and $T = T_\gamma = (T_{\gamma,1}, T_{\gamma,2})$ where $T = T_0 = (g^{c_1 c_2 c_3} f^{z_3}, g^{z_3})$ or $T = T_1 = (g^d f^{z_3}, g^{z_3})$. \mathcal{B} first chooses random values $a_2, a_3 \in \mathbb{Z}_p$ and sets

$$\begin{aligned} g^{\vec{b}_{1,1}} &= (g, 1, f), \quad g^{\vec{b}_{1,2}} = (g, g^{a_2}, 1), \quad g^{\vec{b}_2} = (g^{a_2}, g^{-1}, g^{a_3}), \quad g^{\vec{b}_3} = (f, f^{a_2} g^{-a_3}, g^{-1}), \\ (g^{\vec{b}_{1,2}})^{c'_1} (g^{\vec{b}_3})^{z_1} &= (g^{c_1 c_2} f^{z_1}, (g^{c_1 c_2} f^{z_1})^{a_2} (g^{z_1})^{-a_3}, (g^{z_1})^{-1}), \\ (g^{\vec{b}_{1,2}})^{c'_2} (g^{\vec{b}_3})^{z_2} &= (g^{c_3} f^{z_2}, (g^{c_3} f^{z_2})^{a_2} (g^{z_2})^{-a_3}, (g^{z_2})^{-1}), \\ T' &= (T_{\gamma,1}, (T_{\gamma,1})^{a_2} (T_{\gamma,2})^{-a_3}, (T_{\gamma,2})^{-1}). \end{aligned}$$

Intuitively, it sets $a'_1 = \log f, a'_2 = a_2, a'_3 = a_1 a_2 - a_3$ and $c'_1 = c_1 c_2, c'_2 = c_3$ where a'_1, a'_2, a'_3 are elements of basis vectors for the Assumption 6-4. Next, it gives the tuple $D' = ((p, \mathbb{G}, \mathbb{G}_T, e), g^{\vec{b}_{1,1}}, g^{\vec{b}_{1,2}}, g^{\vec{b}_2}, g^{\vec{b}_3}, (g^{\vec{b}_{1,1}})^{c'_1} (g^{\vec{b}_2})^{z_1}, (g^{\vec{b}_{1,1}})^{c'_2} (g^{\vec{b}_2})^{z_2})$ and T' to \mathcal{A} . Then \mathcal{A} outputs a guess γ' . \mathcal{B} also outputs γ' . If the advantage of \mathcal{A} is ε , then the advantage of \mathcal{B} is greater than ε since the distribution of the challenge tuple to \mathcal{A} is equal to the Assumption 6-4. \square

7 Conclusion

We converted the HVE scheme of Boneh and Waters, the delegatable HVE scheme of Shi and Waters, and the efficient HVE scheme of Lee and Lee from composite order bilinear groups to prime order bilinear groups. Though we used our conversion method to HVE schemes that based on the decisional C3DH assumption, it would be possible to use our method to other scheme in composite order bilinear groups that based on the decisional C3DH assumption.

References

- [1] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.
- [2] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.
- [3] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004.
- [4] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.
- [5] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In Joe Kilian, editor, *Theory of Cryptography - TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2005.
- [6] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *Theory of Cryptography - TCC 2011*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273. Springer, 2011.
- [7] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, *Theory of Cryptography - TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, 2007.
- [8] Léo Ducas. Anonymity from asymmetry: New constructions for anonymous hibe. In Josef Pieprzyk, editor, *Topics in Cryptology - CT-RSA 2010*, volume 5985 of *Lecture Notes in Computer Science*, pages 148–164. Springer, 2010.
- [9] David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 44–61. Springer, 2010.

- [10] Sanjam Garg, Abishek Kumarasubramanian, Amit Sahai, and Brent Waters. Building efficient fully collusion-resilient traitor tracing and revocation schemes. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM Conference on Computer and Communications Security - CCS 2010*, pages 121–130. ACM, 2010.
- [11] Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In Yuliang Zheng, editor, *Advances in Cryptology - ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.
- [12] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security - CCS 2006*, pages 89–98. ACM, 2006.
- [13] Vincenzo Iovino and Giuseppe Persiano. Hidden-vector encryption with groups of prime order. In Steven D. Galbraith and Kenneth G. Paterson, editors, *Pairing-Based Cryptography - Pairing 2008*, volume 5209 of *Lecture Notes in Computer Science*, pages 75–88. Springer, 2008.
- [14] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.
- [15] Jonathan Katz and Arkady Yerukhimovich. On black-box constructions of predicate encryption from trapdoor permutations. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 197–213. Springer, 2009.
- [16] Kwangsu Lee and Dong Hoon Lee. Improved hidden vector encryption with short ciphertexts and tokens. *Des. Codes Cryptography*, 58(3):297–319, 2011.
- [17] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In Daniele Micciancio, editor, *Theory of Cryptography - TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer, 2010.
- [18] Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 214–231. Springer, 2009.
- [19] Jong Hwan Park. Efficient hidden vector encryption for conjunctive queries on encrypted data. *IEEE Trans. Knowl. Data Eng.*, 23(10):1483–1497, 2011.
- [20] Jong Hwan Park. Inner-product encryption under standard assumptions. *Des. Codes Cryptography*, 58(3):235–257, 2011.
- [21] Elaine Shi, John Bethencourt, Hubert T.-H. Chan, Dawn Xiaodong Song, and Adrian Perrig. Multi-dimensional range query over encrypted data. In *IEEE Symposium on Security and Privacy*, pages 350–364. IEEE Computer Society, 2007.
- [22] Elaine Shi and Brent Waters. Delegating capabilities in predicate encryption systems. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfssdóttir, and Igor Walukiewicz, editors, *ICALP 2008*, volume 5126 of *Lecture Notes in Computer Science*, pages 560–578. Springer, 2008.

- [23] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology - EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer, 1997.
- [24] Brent Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.
- [25] Brent R. Waters, Dirk Balfanz, Glenn Durfee, and Diana K. Smetters. Building an encrypted and searchable audit log. In *NDSS 2004*. The Internet Society, 2004.

A Generic Group Model

In this section, we show that the P3DH assumption holds in the generic group model. The generic group model introduced by Shoup [23] is a tool for analyzing generic algorithms that work independently of the group representation.

A.1 Master Theorem

We generalize the master theorem of Katz et al. [14] to use prime order bilinear groups instead of composite order bilinear groups and to use multiple groups elements in the target instead of just one element.

Let \mathbb{G}, \mathbb{G}_T be cyclic bilinear groups of order p where p is a large prime. The bilinear map is defined as $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. In the generic group model, a random group element of \mathbb{G}, \mathbb{G}_T is represented as a random variable P_i, R_i respectively where P_i, R_i are chosen uniformly in \mathbb{Z}_p . We say that a random variable has degree t if the maximum degree of any variable is t . Then we can naturally define the dependence and independence of random variables as in Definition A.1.

Definition A.1. Let $P = \{P_1, \dots, P_u\}$, $T_0 = \{T_{0,1}, \dots, T_{0,m}\}$, $T_1 = \{T_{1,1}, \dots, T_{1,m}\}$ be random variables over \mathbb{G} where $T_{0,i} \neq T_{1,i}$ for all $1 \leq i \leq m$, and let $R = \{R_1, \dots, R_v\}$ be random variables over \mathbb{G}_T . We say that T_b is dependent on A if there exists constants $\{\alpha_i\}, \{\beta_i\}$ such that

$$\sum_i^m \alpha_i T_{b,i} = \sum_i^u \beta_i \cdot P_i$$

where $\alpha_i \neq 0$ for at least one i . We say that T_b is independent of P if T_b is not dependent on P .

Let $S_1 = \{(i, j) \mid e(T_{0,i}, T_{0,j}) \neq e(T_{1,i}, T_{1,j})\}$ and $S_2 = \{(i, j) \mid e(T_{0,i}, P_j) \neq e(T_{1,i}, P_j)\}$. We say that $\{e(T_{b,i}, T_{b,j})\}_{(i,j) \in S_1} \cup \{e(T_{b,i}, P_j)\}_{(i,j) \in S_2}$ is dependent on $P \cup R \cup \{e(T_{b,i}, T_{b,j})\}_{(i,j) \notin S_1} \cup \{e(T_{b,i}, P_j)\}_{(i,j) \notin S_2}$ if there exist constants $\{\alpha_{i,j}\}, \{\alpha'_{i,j}\}, \{\beta_{i,j}\}, \{\beta'_{i,j}\}, \{\gamma_{i,j}\}, \{\delta_i\}$ such that

$$\begin{aligned} & \sum_{(i,j) \in S_1} \alpha_{i,j} \cdot e(T_{b,i}, T_{b,j}) + \sum_{(i,j) \notin S_1} \alpha'_{i,j} \cdot e(T_{b,i}, T_{b,j}) + \sum_{(i,j) \in S_2} \beta_{i,j} \cdot e(T_{b,i}, P_j) + \sum_{(i,j) \notin S_2} \beta'_{i,j} \cdot e(T_{b,i}, P_j) \\ &= \sum_i^u \sum_j^u \gamma_{i,j} \cdot e(P_i, P_j) + \sum_i^v \delta_i \cdot R_i. \end{aligned}$$

where $\alpha_{i,j} \neq 0$ for at least one $(i, j) \in S_1$ or $\beta_{i,j} \neq 0$ for at least one $(i, j) \in S_2$. We say that $\{e(T_{b,i}, T_{b,j})\}_{(i,j) \in S_1} \cup \{e(T_{b,i}, P_j)\}_{(i,j) \in S_2}$ is independent of $P \cup R \cup \{e(T_{b,i}, T_{b,j})\}_{(i,j) \notin S_1} \cup \{e(T_{b,i}, P_j)\}_{(i,j) \notin S_2}$ if $\{e(T_{b,i}, T_{b,j})\}_{(i,j) \in S_1} \cup \{e(T_{b,i}, P_j)\}_{(i,j) \in S_2}$ is not dependent on $P \cup R \cup \{e(T_{b,i}, T_{b,j})\}_{(i,j) \notin S_1} \cup \{e(T_{b,i}, P_j)\}_{(i,j) \notin S_2}$.

Using the above dependence and independence of random variables, we can obtain the following theorem from the master theorem of Katz et al. [14].

Theorem A.1. *Let $P = \{P_1, \dots, P_u\}$, $T_0 = \{T_{0,1}, \dots, T_{0,m}\}$, $T_1 = \{T_{1,1}, \dots, T_{1,m}\}$ be random variables over \mathbb{G} where $T_{0,i} \neq T_{1,i}$ for all $1 \leq i \leq m$, and let $R = \{R_1, \dots, R_v\}$ be random variables over \mathbb{G}_T . Consider the following experiment in the generic group model:*

An algorithm is given $P = \{P_1, \dots, P_u\}$ and $R = \{R_1, \dots, R_v\}$. A random bit b is chosen, and the adversary is given $T_b = \{T_{b,1}, \dots, T_{b,m}\}$. The algorithm outputs a bit b' , and succeeds if $b' = b$. The algorithm's advantage is the absolute value of the difference between its success probability and $1/2$.

Let $S_1 = \{(i, j) \mid e(T_{0,i}, T_{0,j}) \neq e(T_{1,i}, T_{1,j})\}$ and $S_2 = \{(i, j) \mid e(T_{0,i}, P_j) \neq e(T_{1,i}, P_j)\}$. If T_b is independent of P for all $b \in \{0, 1\}$, and $\{e(T_{b,i}, T_{b,j})\}_{(i,j) \in S_1} \cup \{e(T_{b,i}, P_j)\}_{(i,j) \in S_2}$ is independent of $P \cup R \cup \{e(T_{b,i}, T_{b,j})\}_{(i,j) \notin S_1} \cup \{e(T_{b,i}, P_j)\}_{(i,j) \notin S_2}$ for all $b \in \{0, 1\}$, then any algorithm \mathcal{A} issuing at most q instructions has an advantage at most $O(q^2 t/p)$.

Note that this theorem that is a slight modification of that of Katz et al. [14] still holds in prime order bilinear groups since the dependent equation of an adversary can be used to distinguish the target T_b of the assumption. Additionally, it still holds when the target consists of multiple group elements since the adversary can only make a dependent equation in Definition A.1.

A.2 Analysis of P3DH Assumption

To analyze the P3DH assumption in the generic group model, we only need to show the independence of T_0, T_1 random variables. Using the notation of previous section, the P3DH assumption can be written as follows

$$P = \{1, X, A, XA, B, XB, AB + XZ_1, Z_1, C + XZ_2, Z_2\}, R = \{1\}$$

$$T_0 = \{ABC + XZ_3, Z_3\}, T_1 = \{D + XZ_3, Z_3\}.$$

The T_1 has a random variable D that does not exist in P . Thus the independence of T_1 is easily obtained. Therefore, we only need to consider the independence of T_0 . First, T_0 is independent of P since T_0 contains Z_3 that does not exist in P . For the independence of $\{e(T_{0,i}, T_{0,j})\}_{(i,j) \in S_1} \cup \{e(T_{0,i}, P_j)\}_{(i,j) \in S_2}$, we should define two sets S_1, S_2 . We obtain that $S_1 = \{(1, 1), (1, 2), (2, 1), (2, 2)\}$. However, $e(T_{0,i}, T_{0,j})$ contains Z_3^2 because of Z_3 in T_0 , and Z_3^2 can not be obtained from the right part of the equation in Definition A.1. Thus, the constants $\alpha_{i,j}$ should be zero for all (i, j) . From this, we obtain the simple equations as follows

$$\sum_{(i,j) \in S_2} \beta_{i,j} \cdot e(T_{b,i}, P_j) + \sum_{(i,j) \notin S_2} \beta'_{i,j} \cdot e(T_{b,i}, P_j) = \sum_i^u \sum_j^u \gamma_{i,j} \cdot e(P_i, P_j) + \sum_i^v \delta_i \cdot R_i.$$

The set S_2 is defined as $\{(i, j) \mid \forall i, j\}$ because of D in T_1 . However, Z_3 in T_0 should be removed to construct a dependent equation since Z_3 does not exist in P, R . To remove Z_3 from the left part of the above simple equation, two random variables Y, XY should be paired with $T_{0,i}$ for some $Y \in P$. If Z_3 is removed in the left part of the above simple equation, then the left part has at least a degree 3 and it contains ABC . To have a degree 3 in the right part of the above simple equation, $AB + XZ_1, Z_1$ should be used. However, the right part of the above equation can not contain ABC since C, XC do not exist in P . Therefore, the independence of T_0 is obtained.